

INFORMÁTICA | CONHECIMENTO | DESENVOLVIMENTO

36

GRAVAÇÃO DE CDs

Guia completo para queimar mídias em uma máquina rodando Linux

36

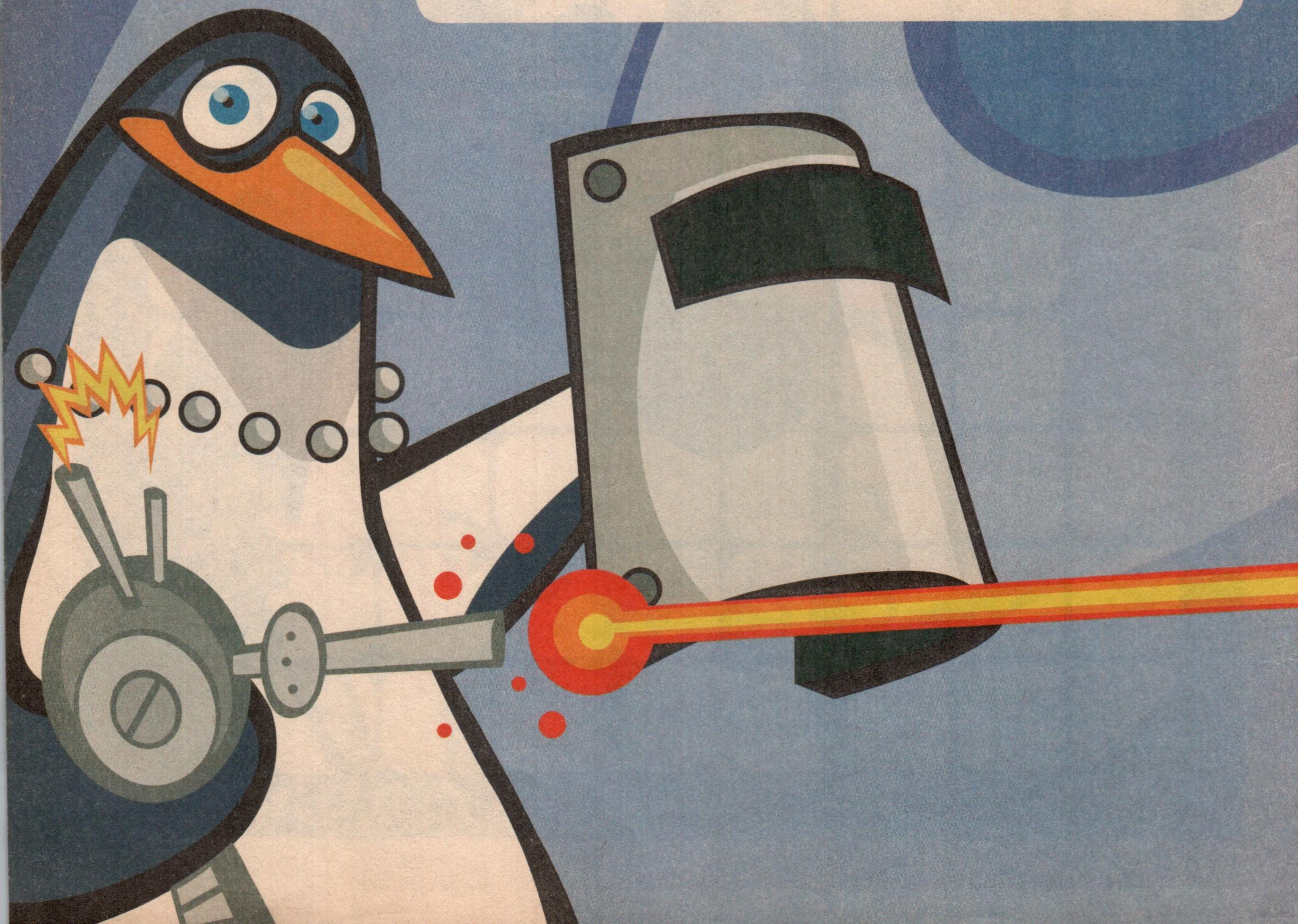
GRAVAÇÃO DE CDs no LINUX

Aprenda a queimar mídias de áudio e de dados no sistema operacional do pingüim

POR JOSÉ ANTONIO DA SILVA NETO

Mais de uma década depois de começar sua revolução, o Linux já pode ser considerado um sistema operacional maduro. Hoje, ele é capaz de realizar com alta qualidade a maioria das tarefas que o sistema do tio Bill faz.

Nas próximas páginas do ALT, mostraremos como manipular CDs no Linux. Entre outros tópicos, abordaremos a gravação de CDs de dados – bootáveis ou não –, de CDs de áudio, sem se esquecer, é claro, da ripagem de CDs. Os testes foram realizados no laboratório da redação com uma máquina rodando o Slackware 9.1, por meio de linhas de comandos e gerenciadores gráficos.



APLICATIVOS DE LINHA DE COMANDO

Os sistemas Unix livres, como as versões do Linux e do BSD, costumam oferecer um conjunto de ferramentas completo para a criação de CDs. Entre eles, destacam-se o Mkisofs, o Cdrecord e o Cdrdao, que apresentaremos agora:

MKISOFS - geralmente, para gravar um CD no Linux, é preciso criar uma "imagem ISO", que é uma cópia fiel dos arquivos a serem gravados no padrão internacional ISO 9660. Essa imagem pode ser de um simples CD de dados ou de um CD bootável, que é inicializado durante o boot pela BIOS. Vamos mostrar a sintaxe básica dos dois casos:

CD comum de dados

Suponha que criamos um subdiretório chamado **teste** no diretório corrente e que movemos para lá todos os arquivos. Uma possível sintaxe para criar uma imagem ISO contendo os arquivos do diretório **teste** seria:

mkisofs -v -r -J -l -L -T -o minha_imagem.iso teste/

(mkisofs = make iso file system = criar um sistema de arquivo ISO)

Seguem as imagens capturadas na criação da imagem ISO:

As chaves são indicações para que sejam geradas permissões e arquivos adicionais em relação aos arquivos que se encontram no diretório-teste.

- **-V:** um padrão do mundo Unix/Linux (v de verbose). Com esta opção, o aplicativo emite mensagens sobre o andamento da criação da imagem ISO
- **-r:** permite acesso público para sistemas não Unix
- **-J:** gera o reconhecimento de nomes longos (que ultrapassem os 8 caracteres permitidos pelo rigoroso ISO 9660) pelo Windows (extensões Joliet)
- **-l:** possibilita o reconhecimento de nomes com até 31 caracteres
- **-L:** permite espaços no ISO 9660
- **-T:** cria recursivamente, diretório por diretório, um arquivo texto chamado TRANS.TBL, que lista o conteúdo do diretório
- **-o:** necessário para fornecer um nome a uma determinada imagem ISO

Há muitas outras opções nas páginas de manual do Cdrecord, que podem ser conferidas no seu manual deste aplicativo.

Como no Unix tudo é tratado como um arquivo, não é de se estranhar que podemos montar uma imagem ISO!

Com isso, podemos montar o nosso exemplo (não se esqueça de mudar seu status de usuário para root antes!):

mount -o loop minha_image.iso /mnt/imagens

Se esse recurso não funcionar, tente utilizar a seguinte linha de comando:

mount -t iso9660 -o loop minha_image.iso /mnt/imagens

Agora, basta acessar o diretório **/mnt/imagens**, e manipular os arquivos montados no modo "apenas leitura".

Isso é preciso porque, infelizmente, o Unix ainda não suporta a edição de imagens ISO de um modo simples. Como "simples", entendemos a impossibilidade de edição da imagem ISO.

No decorrer do tutorial, apresentaremos o KISO, um aplicativo do KDE que, embora esteja no começo do seu

desenvolvimento, já permite manipulações interessantes das imagens.

Para desmontar, bastar sair do diretório e digitar o seguinte comando:

```
umount /mnt/imagens
```

Abaixo segue uma imagem mostrando o ISO montado:

3

```
Terminal
o  Editar  Ver  Terminal  Ir  Ajuda
05b# mount -o loop minha_imagem.iso /mnt/imagens/
05b# cd /mnt/imagens/
05b# ls -l
9
r--    1 root    root     14336 2004-05-17 15:02 cds_no_tux.doc
r-x    6 root    root      2048 2004-05-17 14:41 OpenOffice.orgl.1.0
r-x    4 root    root      2048 2004-05-17 15:04 rr_moved
r--    1 root    root       117 2004-05-17 15:04 TRANS.TBL
05b#
```

CDs bootáveis

Sim! Podemos criar CDs bootáveis no Linux usando a linha de comando pura. Acreditamos que, antes de apresentar uma definição formal do processo, devemos ilustrá-lo através de exemplos práticos:

CD bootável do OpenBSD

Embora as imagens ISO oficiais do OpenBSD não estejam disponíveis livremente para download, é possível criá-las. Porém, é importante comprar as caixinhas oficiais do projeto, sua principal fonte de fomento.

Para criar a imagem, siga os procedimentos abaixo:

• Crie a seqüência de diretórios:

```
mkdir Puffy
mkdir Puffy/3.5
mkdir Puffy/3.5/i386
mkdir Puffy/3.5/packages
```

("Puffy" é o nome do mascote do OpenBSD: um baiacu, peixe extremamente venenoso...)

• Rode um cliente de ftp (o gftp, por exemplo), acesse o servidor de ftp do OpenBSD (clique na aba **Marcadores, BSD Sites, OpenBSD**. Depois que o servidor for devidamente acessado, faça o seguinte:

• Baixe os arquivos: **XF4.tar.gz, ports.tar.gz, src.tar.gz** e **sys.tar.gz** (do diretório **/pub/OpenBSD/3.5**) e depois mova-os para o path **Puffy/3.5**

• Baixe todo o conteúdo do diretório **/pub/OpenBSD/3.5/i386** e mova-o para o path **Puffy/3.5/i386**

• O diretório packages que acabamos de criar é voltado para a inclusão opcional de pacotes (de até 781 MB em um CD de 90 minutos). Basta acessar o diretório **/pub/OpenBSD/3.5/packages** e baixar o que quiser. Depois mova-o para **Puffy/3.5/packages.**

A referência básica para esse exemplo se encontra em **http://www.shockley.net/obsd-bootcd.asp**

Criando o ISO bootável

```
mkisofs -b 3.5/i386/cdrom35.fs -v -r -J -l -L -T -V
"OpenBSD-3.5" -A "OpenBSD v3.5" -c boot.catalog -o
OpenBSD35.iso Puffy/
```

Em relação ao exemplo do CD comum de dados, para o CD bootável, temos as seguintes chaves adicionais:

• **-b :** informamos ao Mkisofs que se trata de um CD bootável seguindo o padrão "El Torito" de CDs bootáveis.

• **-c :** necessário para que o catálogo de boot seja criado e incluído no primeiro setor da imagem de CD.

• **-V :** especifica o volume ID a ser escrito no setor mestre.

• **-A:** adiciona uma nota a respeito da imagem ISO.

Depois é só gravar (ainda vamos mostrar como se faz isso) e bootar!

CD bootável do Slackware

Do mesmo modo que no exemplo anterior, é preciso baixar os arquivos necessários. Nos testes, criamos uma imagem bootável com todo o conteúdo da distribuição Slackware, que ocupou cerca de 2.8 GB.

Devido ao tamanho, trata-se de uma imagem ISO para ser gravada em DVD. Boa parte do conteúdo desse tutorial também vale para o caso de gravação de DVDs.

Para criarmos uma imagem da árvore de diretórios do Slackware 10, utilizamos o aplicativo **wget**, que vem instalado na maioria esmagadora das distribuições Linux. Em nossos testes, utilizamos o wget conforme a linha de comando abaixo:

```
wget -v -c -r ftp://inferno.bioinformatics.vt.edu/
linux-distros/slackware/slackware-10.0/
```

O comando citado clonará a árvore de diretórios. O link utilizado (um repositório ftp) é apenas um exemplo. Toda a árvore do Slack 10 será gravada no diretório em que foi disparado o comando. Antes de testar a opção, é aconselhável criar um diretório e usar o comando a partir dele.

Para que a imagem caiba num CD usual (o sistema completo demandaria um DVD, como já dissemos), devemos limitar os arquivos a serem incluídos na imagem ISO. Novamente, vamos usar o Mkisofs para criar a imagem ISO. Para isso, abra um terminal e digite:

```
mkisofs -o /caminho/onde/a/imagem/sera/armazenada
slackware.iso \

-R -J -V "Slackware Install" \
-x ./bootdisks \
-x ./extra \
-x ./slackware/gnome \
-x ./pasture \
-x ./rootdisks \
-x ./source \
-x ./zipslack \
-hide-rr-moved \
-v -d -N -no-emul-boot -boot-load-size 4 -boot-info-
table \
-sort isolinux/iso.sort \
-b isolinux/isolinux.bin \
-c isolinux/isolinux.boot \
-A "Slackware Install CD" .
```

Há algumas diferenças em relação ao exemplo anterior:

- **-V:** permite associar um nome de volume ao setor de boot do CD
- **-x:** informa ao Mkisofs para excluir os paths correspondentes
- **-d:** omite períodos de rastreamento de arquivos que não possuem um período
- **-N:** omite os números de versões dos nomes de arquivos ISO 9660
- **-no-emul-boot:** informa que a imagem de boot usada para criar a ISO é "sem emulação". A BIOS vai bootar o CD sem realizar nenhuma emulação de disco
- **-hide-rr-move:** renomeia o diretório RR_MOVED para .rr_moved na árvore Rock Ridge
- **-sort:** ordena as localizações dos arquivos na mídia
- **-A:** especifica um texto que será escrito no cabeçalho do volume
- **. : o ponto "."** indica que a fonte dos arquivos é o diretório corrente que, no caso, é o diretório que contém a raiz da árvore de diretórios do Slackware. Em nossso exemplo, foi **/teste/inferno.bioinformatics.vt.edu/linux-distros/slackware/slackware-10.0/**

A referência básica para esse exemplo é o arquivo **README,** que acompanha a distro no diretório **isolinux** do CD1.

CD bootável do Win98

Ainda hoje, o Windows 98 é muito utilizado e, muitas vezes, é fundamental ter um disquete de boot para tentar salvar o sistema. Que tal criar um CD bootável com o disquete de inicialização do Windows 98? É muito simples fazer isso no Linux usando as mesmas técnicas apresentadas nos exemplos anteriores.

Tendo o disquete de boot do Windows 98 em mãos, crie uma imagem dele utilizando o utilitário dd do Linux. Ele faz cópias bit-a-bit de uma mídia e, por isso, é independente do sistema de arquivos.

Acessando o dd

Para acessar o **dd**, coloque o disquete no drive, abra um terminal e digite:

```
dd if=/dev/fd0 of=Win98.img &
```

A extensão "img" é imaterial no mundo Unix. Poderíamos ter usado qualquer outra extensão. Como no caso de uma imagem ISO, podemos montar o Win98.img:

```
mount -o loop Win98.img /mnt/imagens
```

Agora vamos criar alguns diretórios:

```
mkdir Windows
mkdir Windows/boot
mkdir Windows/pacotes
```

Em seguida, é preciso mover a imagem Win98.img para o diretório Windows/boot, que é para onde vamos apontar a imagem bootável no Mkisofs.

Já no path Windows/pacotes, podemos acrescentar qualquer aplicativo que rode em ambiente DOS.

Criando a imagem bootável:

```
mkisofs -v -r -J -l -L -T -b boot/Win98.img -c boot.cata-
log-o win.iso Windows/
```

Pronto. Agora podemos criar uma mídia de inicialização (um CD!) muito mais rápida que um disquete.

A generalização é válida em algumas situações:

```
mkisofs -v -r -J -l -L -T -b ramo_da_arvore/imagem_de_boot.img -c boot.catalog \
-o minha_imagem.iso diretorio_dos_arquivos/
```

GRAVANDO CDS

Agora que já sabemos criar algumas imagens ISO, como gravá-las nos CDs? É aqui que entra o Cdrecord. Com este programa, podemos gravar CDs facilmente via linha de comando.

Mas atenção: há pequenas variações na sintaxe em relação aos kernels da família 2.4 e 2.6. Até as versões de kernel das famílias 2.4, é usada uma camada de emulação SCSI para o controle de gravadores de CD-R/RW.

Em distros como Mandrake e SuSE, essa adaptação é feita automaticamente pelo instalador, pois em geral as pessoas possuem gravadores IDE.

No Slackware, o instalador requisita essa informação durante a instalação. Uma entrada do tipo **append="hdc=ide-scsi"** (indicando que temos um gravador de CDs como mestre na segunda IDE) é adicionada ao Lilo. Com isso, o kernel irá carregar a emulação SCSI durante o boot.

Mas na nova família estável de kernels, a 2.6, isso não é mais necessário. Basta indicar a identificação do dispositivo IDE diretamente (com **/dev/hdc,** por exemplo):

SINTAXE DO CDRECORD NA FAMÍLIA 2.4

```
cdrecord -v -fs=50M speed=32 dev=0,0,0 -eject -data minha_imagem.iso
```

- **-v:** modo verbose, para que o sistema informe sobre o andamento da gravação

- **-fs:** memória adicional que liberamos para que o Cdrecord a utilize. O mínimo default são 4 MB. Se omitirmos a chave -fs, usamos 50 MB, mas se dispomos de pouca memória RAM, é possível utilizar uma quantidade menor. Quanto mais memória, menor é a chance de ocorrerem falhas, principalmente em velocidades de gravação altas.

- **speed:** aqui setamos a velocidade de gravação desejada. Em nosso exemplo, usamos 32X (4800 Kb/s), mas a velocidade adequada deve ser calibrada de acordo com os recursos da sua máquina (com 256 MB de RAM e um Athlon1300+, podemos tranqüilamente usar 32X).

- **dev :** neste caso, devemos entrar com a identificação do dispositivo (o "0,0,0"), mas o número correto para uma dada máquina deve ser obtido usando o próprio cdrecord através da seguinte linha de comando:

```
cdrecord -scanbus
```

- **-eject:** informa ao Cdrecord para ejetar o CD após a gravação

- **-data :** informa ao Cdrecord que se trata de um CD de dados

- **minha_imagem.iso:** a imagem a ser gravada

SINTAXE DO CDRECORD NA FAMÍLIA 2.6

Como não temos a camada de emulação SCSI (embora ainda seja possível ativá-la), a única mudança na sintaxe se localiza, é claro, na identificação do dispositivo:

```
cdrecord -v -fs=50M speed=32 dev=/dev/hdc -data minha_imagem.iso
```

Ou seja, não precisamos mais do número "0,0,0". Em vez disso, setamos diretamente o dispositivo IDE.

IDENTIFICANDO A QUALIDADE DA MÍDIA

Um fato pouco conhecido é que podemos ler no Linux os dados do fabricante contidos num CD-R (no Windows, podemos usar o **cdridentifier**). São informações importantes, como o nível de potência a ser aplicado pelo canhão laser do gravador, a substância química utilizada na fabricação do CD, a qualidade da substância e o fabricante real da mídia).

Para fazer isso na prática, basta colocar um CD (virgem ou não!) no gravador, abrir um terminal e digitar (como root):

• família 2.4.*:
cdrecord -atip dev=0,0,0

• família 2.6.*:
cdrecord -atip dev=/dev/hdc

Vale lembrar que são apenas exemplos. O leitor precisa fazer algumas adaptações para ficar de acordo com as configurações do seu hardware.

Testamos a mídia Verbatim e, como mostra a próxima imagem, trata-se de uma mídia de excelente qualidade (A+ e long strategy – alto padrão, segundo os fabricantes da Mitsubishi Chemical Corporation).

4

```
Revision       : 'QYS1'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags   : MMC-3 SWABAUDIO BURNFREE FORCESPEED
Supported modes: TAO PACKET SAO SAO/R96P SAO/R96R RAW/R16 RAW/R96P RAW/
ATIP info from disk:
  Indicated writing power: 4
  Is not unrestricted
  Is not erasable
  Disk sub type: Medium Type A, high Beta category (A+) (3)
  ATIP start of lead in:  -11077 (97:34/23)
  ATIP start of lead out: 359848 (79:59/73)
Disk type:    Long strategy type (Cyanine, AZO or similar)
Manuf. index: 11
Manufacturer: Mitsubishi Chemical Corporation
bash-2.05b$
```

Aconselhamos sempre fazer esse tipo de teste na hora de comprar uma mídia CD-R, evitando levar gato por lebre.

USANDO O CDRDAO

Além do formato internacional ISO, existe um outro formato de imagem de CDs muito usado no mundo DOS/Windows: o bin/cue. Enquanto o **".bin"** é a imagem propriamente dita, o **".cue"** é um arquivo descritor associado. Como o Cdrecord não suporta a gravação desse tipo de arquivo, uma alternativa é usar o Cdrdao:

GRAVANDO UMA IMAGEM BIN/CUE

Este exemplo também serve para CDs de PlayStation

cdrdao write --buffers 64 --device 0,0,0 --speed 16 nome_do_arquivo.cue

Onde:

- **write:** grava no CD-R

- **--buffers:** com esta opção, fixamos o número de buffers que deixamos disponíveis para o cdrdao, visto que o default é 32

- **--device:** é o número do dispositivo de gravação
- **--speed:** nesta opção, definimos a velocidade

BIN2ISO

Como observamos na descrição do Cdrdao, podemos também gravar imagens bin/cue no Linux. Mas e se quiséssemos manipulá-las? Diretamente não seria possível. Apesar disso, podemos converter imagens bin/cue para ISO usando o **bin2iso**, um aplicativo desenvolvido por BobDoiron (**http://users.andara.com/~doiron/bin2iso/**). O bin2iso consegue funcionar até quando não dispomos do arquivo descritor cue.

- ***Exemplo1 - :***

bin2iso nome_do_arquivo.cue

- ***Exemplo2*** **- (caso em que só temos o .bin):**

bin2iso nome_do_arquivo.cue -c nome_arquivo.bin

OBS: por ser um pacote um pouco difícil encontrar (o link do pacote para Linux está quebrado!), baixamos uma fonte em .rpm, geramos pacotes binários nos formatos .rpm e .tgz e incluímos tudo no CD.

RIPANDO COM O CDPARANOIA

Música é, sem dúvida, uma das poucas manifestações realmente universais da alma humana, portanto, dentro do ambiente tecnológico que o computador nos oferece, não poderiam faltar ferramentas para manipulação de CDs de áudio (muito menos no sistema operacional Linux). O **CDParanoia** é um ripador de linha de comando eficiente, com dois modos básicos de extração:

- **Extraindo uma faixa específica**

Abra um terminal e digite:

su (para se tornar root)
cdparanoia -v <número_da_faixa> nome_da_faixa.wav

5

```
Verifying drive can read CDDA...
        Expected command set reads OK.

Table of contents (audio tracks only):
track        length               begin        copy pre ch
===========================================================
  1.    20552 [04:34.02]        0 [00:00.00]    no   no  2
  2.    19285 [04:17.10]    20552 [04:34.02]    no   no  2
  3.    14375 [03:11.50]    39837 [08:51.12]    no   no  2
  4.    16948 [03:45.73]    54212 [12:02.62]    no   no  2
  5.    23745 [05:16.45]    71160 [15:48.60]    no   no  2
  6.    21005 [04:40.05]    94905 [21:05.30]    no   no  2
  7.    20307 [04:30.57]   115910 [25:45.35]    no   no  2
  8.    22323 [04:57.48]   136217 [30:16.17]    no   no  2
  9.    23090 [05:07.65]   158540 [35:13.65]    no   no  2
TOTAL  181630 [40:21.55]    (audio only)

Ripping from sector        0 (track  1 [0:00.00])
          to sector    20551 (track  1 [4:34.01])

outputting to primeira_faixa.wav

 (== PROGRESS == [  +          + +    >        | 014496 00 ] == :-) . ==)
```

A chave **-v** é a usual do mundo Unix. Representa o verbose, que emite mensagens do processo em questão. É interessante notar as "carinhas" (típicas da linguagem de chat) e os sinais que o programa utiliza para simbolizar os eventos durante a **ripagem** (ripar um CD de áudio é extrair suas faixas para o HD).

Por exemplo, o símbolo "+ " significa "perda de streaming ou de outros erros não relatados". Já o emoticon ":-)" indica que a ripagem está ocorrendo normalmente (ao contrário dos avisos pelos "+" que ocorreram no nosso teste).

- Extraindo todas faixas do CD

cdparanoia -v -B

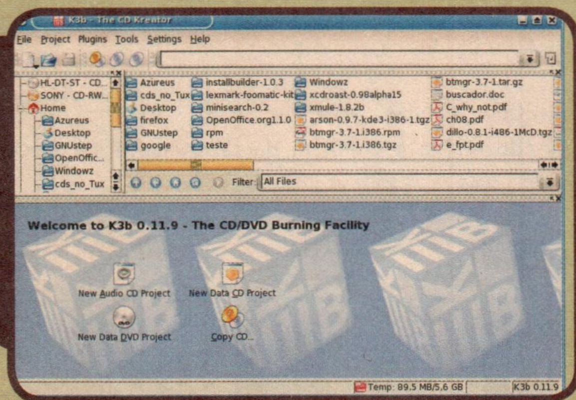
APLICATIVOS GRÁFICOS

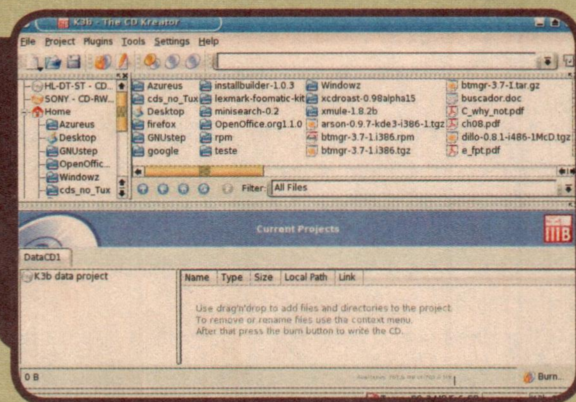
Cada vez mais, o Linux possui aplicativos gráficos de alta qualidade, e a gravação de CDs não poderia ser uma exceção. Vamos apresentar alguns deles: K3B, KISO, Gcombust e XCDroast:

K3B

Embora esteja apenas na versão 0.11, o software K3b, desenvolvido por coders alemães (**http://www.k3b.org/**), já rivaliza com programas como o Nero e o Easy CD Creator do Windows, parecendo-se bastante com o primeiro. A versão que testamos é a 0.11.9. Baixamos o pacote-fonte e compilamos da forma usual, mas o K3b já vem incluído como default em distros mais automatizadas como SuSE e Mandrake.

Agora observemos a tela 6 do nosso passo-a-passo. Um detalhe importante a ser lembrado é que qualquer GUI possui os valores default de configuração, que nem sempre são os mais adequados. No caso do K3b, a primeira coisa que deve ser mudada é a quantidade

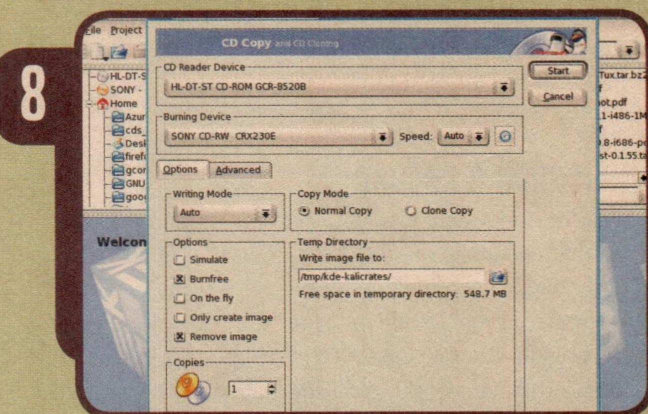




de memória RAM disponível para o cdrecord. Para isso, vá em **Settings, Configure K3B**. Uma janela de configuração será aberta. Depois, acesse **Writing, Advanced**. Finalmente, aumente ao máximo a quantidade de RAM disponível para o cdrecord; esse procedimento evitará falhas de gravação.

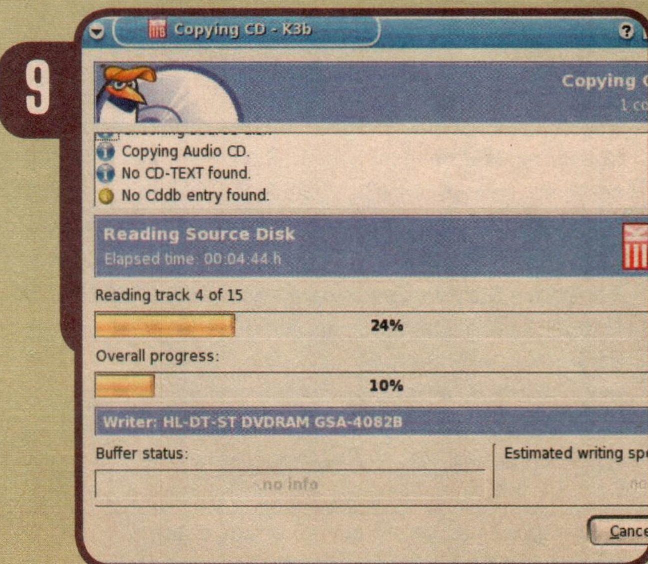
Para criar um projeto de um CD de dados, acesse na seqüência **File, New Project, New Data CD Project**. Uma imagem parecida com a da tela 7 surgirá. Para adicionar arquivos ou diretórios, basta clicar neles com o botão direito do mouse. Surgirá uma tela com a opção **Add to Project**.

Ao terminar de adicionar os arquivos, clique em **K3b data project** e escolha **Burn** para gravar o projeto. Na tela de gravação que irá surgir, escolha em **Advanced** a opção **Multissesion CD,** que não irá "fechar" o CD **(CD Multisessão!)**. Ou seja, se os dados a serem gravados não ocuparem todo o espaço, podemos fazer outras gravações no mesmo CD, até o limite de sua capacidade.

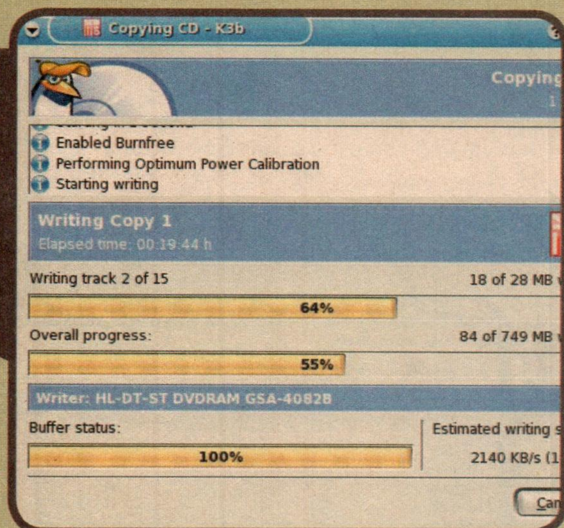


Para fazer cópia de CD para CD, basta ir em **Tools, CD, Copy CD**. Como temos um leitor e um gravador, podemos realizar a leitura ou gravação em um único passo. Note que setamos **Burnfree**. Nesse caso, uma imagem ISO intermediária será gerada no path **/tmp/kde-kalicrates** (podemos mudar o path). O processo de gravação será um pouco lento, mas mais seguro.

As imagens a seguir mostram o processo dividido em duas fases. Devemos frisar que o processo só funcionou graças à utilização de um drive de CD-RW tanto para leitura e gravação.



10



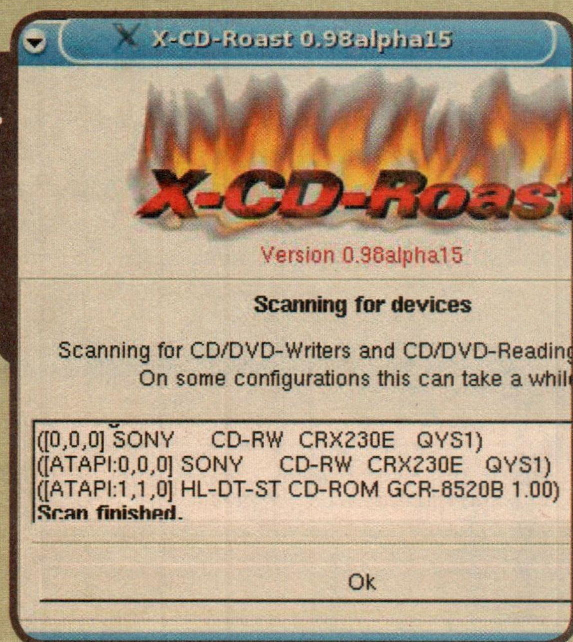
Há muitas outras opções no K3b. Convidamos o nosso caro leitor a testá-las.

XCDROAST

Embora esteja um pouco esquecido, o **XCDRoast** (**www.xcdroast.org**) continua a ser um gerenciador de gravação clássico, no caso de CDs, e com várias funcionalidades. Nos nossos testes, baixamos o pacote mais recente que está disponível e o compilamos.

A seguir, chamaremos o programa de usuário **root**.

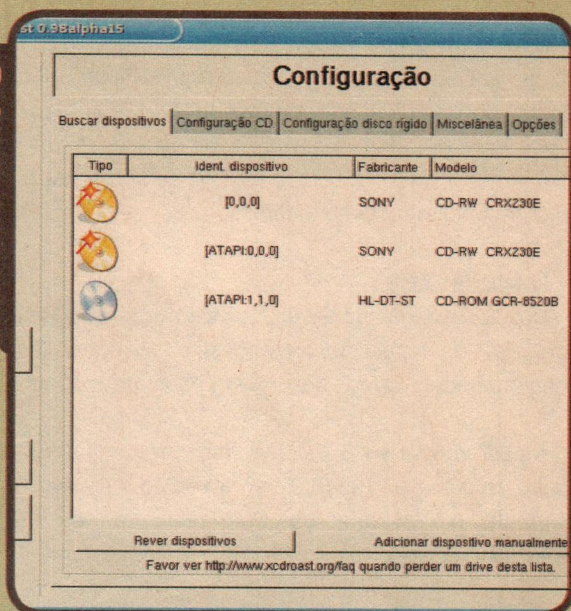
11



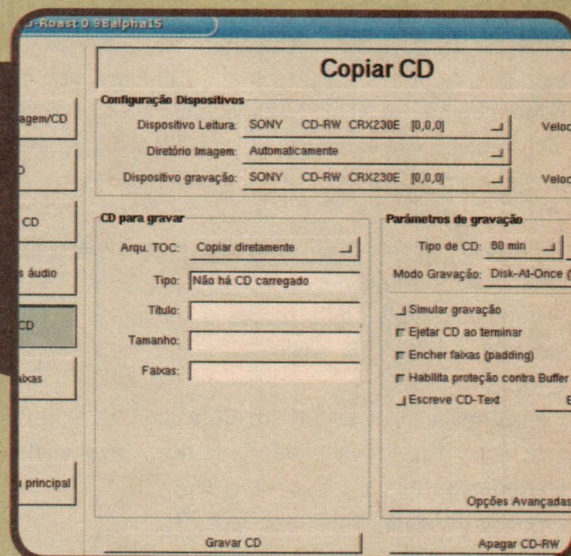
O Xcdroast está procurando os dispositivos de leitura ou gravação presentes no micro.

É importante lembrar que, antes de utilizarmos o Xcdroast, precisamos configurá-lo. O programa se mostrou bem esperto, pois praticamente tudo estava corretamente setado!

12



13



Há muitas opções a serem exploradas. Nesta imagem acima, acessamos o item **Gravar CD**.

Note que o Xcdroast setou a velocidade de gravação no valor máximo suportado pelo gravador, mas sugerimos baixar o valor para 32X.

Lembra-se da opção de ler os dados do fabricante? Pois aqui temos uma entrada para ler diretamente essa informação. Basta clicar em **INFO Atip.**

KISO

Quem já possui um pouco de experiência em gravação de CDs sabe que existem utilitários no Windows que manipulam imagens ISO. Sem dúvida, trata-se de um recurso valioso, pois podemos querer aproveitar uma imagem pronta, simplesmente adicionando ou removendo arquivos e/ou diretórios.

O KISO (**http://kiso.sourceforge.net/**), desenvolvido por Stephan Gans, um coder alemão (na verdade, um cientista da computação), é a primeira opção que surge nesse sentido no mundo Linux. Apesar de estar em sua infância (testamos a versão 0.5.1), o KISO já se apresenta razoavelmente funcional, como veremos.

Instalando o KISO

Nós, da redação da Geek, suavizamos um pouco o processo de instalação, mas, mesmo assim, há algumas dependências. Vamos descrever brevemente quais são:

- **KDE versão >= 3.2:** além desse requisito, foi necessário atualizar o XFree86 para a versão 4.4; isso foi muito simplificado devido ao uso sistemático do swaret (**http://www.swaret.org**)

- **libcddb:** biblioteca escrita na linguagem C (**http://libcddb.sourceforge.net**), que permite acesso a dados de um servidor CDDB (**http://freedb.org/** – um banco de dados para conseguir informações sobre CDs através da internet). Assim como uma busca no banco de dados, retorna informação detalhada sobre um CD específico e permite submeter um novo CD ao BD.

- **libcdio:** pacote GNU (**www.gnu.org/software/libcdio/**) com várias ferramentas para manipulação de CDs, como leitura ou controle do CD e acesso a imagens ISO-9660, além do par BIN/CUE e NRG

- **vcdimager:** pacote GNU para masterizar, extrair e analisar vídeo CDs e super vídeo CDs(**http://www.vcdimager.org/index.phtml**),
  versão utilizada: vcdimager-0.7.20

Para instalar o KISO, praticamente compilamos todos os itens. Não é preciso seguir o mesmo procedimento. Podem existir pacotes compilados nos repositórios da sua distro mas, no caso de não existir um pacote pronto, só compilando mesmo. Em todos os pacotes, a seqüência de instalação foi a usual. Abrimos um terminal e digitamos:

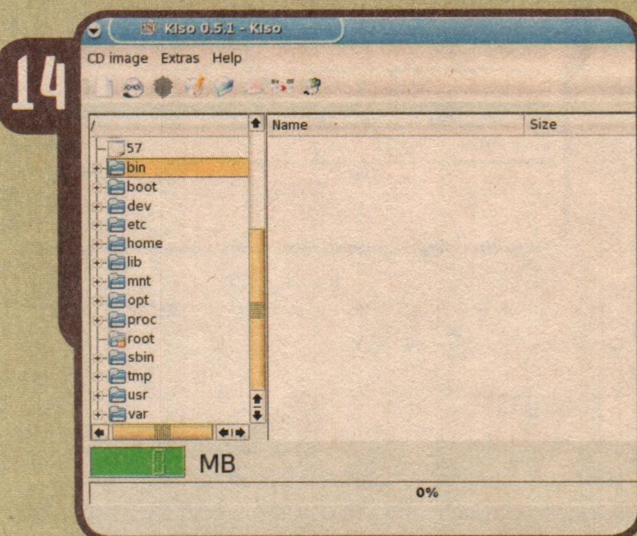
```
./configure
make
su
make install
```

Testando o KISO

Depois que o sistema estiver instalado, basta chamá-lo num terminal, inicialmente como root, pois para montar uma ISO, é preciso ter status de usuário administrador. O Kiso irá requisitar a senha de root e funcionará para qualquer outro usuário do sistema. Portanto, abra um terminal e digite:

```
su
kiso &
```

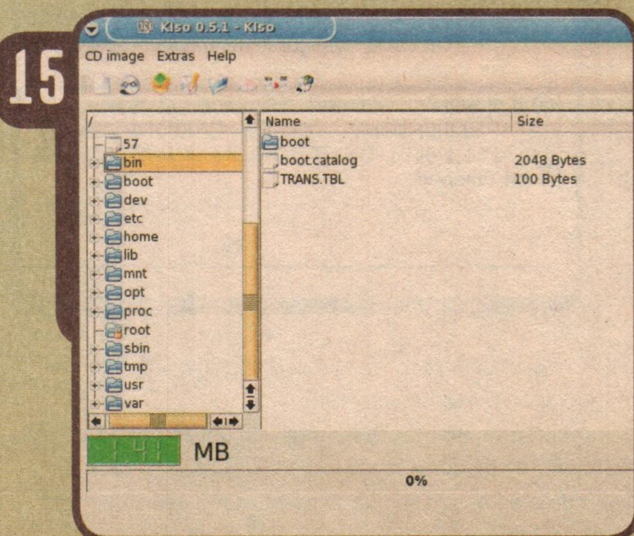
Lembra-se da imagem que criamos (via linha de comando) win.iso? Vamos abri-la usando o KISO. Clique em **CD image** e depois em **Open image**.



Será aberta uma janela para apontarmos a imagem desejada. Aqui, escolhemos **win.iso**.

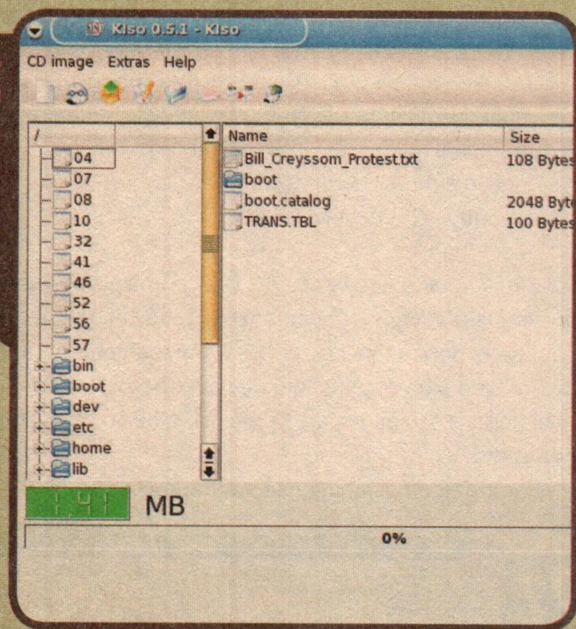
Vemos a seguir a árvore de diretórios contida na imagem, que foi nitidamente preservada:

Vejamos agora se podemos incluir um arquivo no diretório principal da imagem aberta. Vamos adicionar o arquivo "Bill_Creyssom_Protest.txt". Para isso, basta clicar no ícone **Add file** e escolher o arquivo desejado. O resultado da edição está ilustrado abaixo:

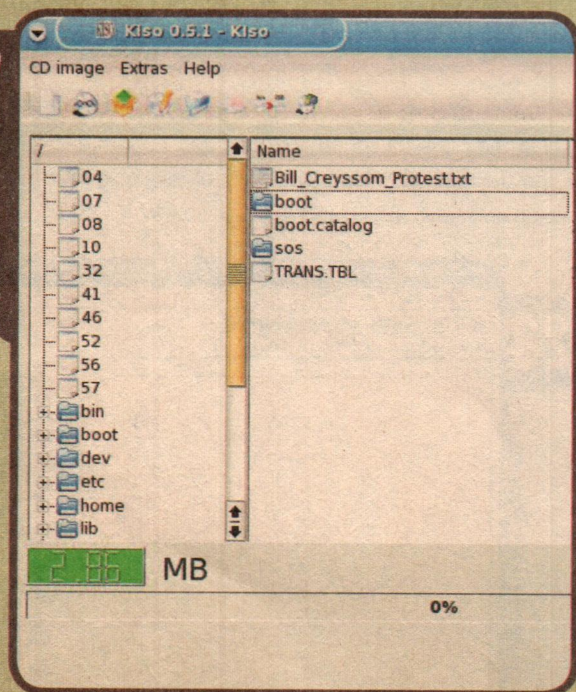


Para salvar a imagem editada, basta clicar no ícone **Salvar imagem** e depois escolher um nome.

O que fizemos em relação a um arquivo pode ser feito em relação a um diretório. Clique com o botão direito na área da GUI na qual estão listados os diretórios/arquivos da imagem aberta e escolha **Add directory**.

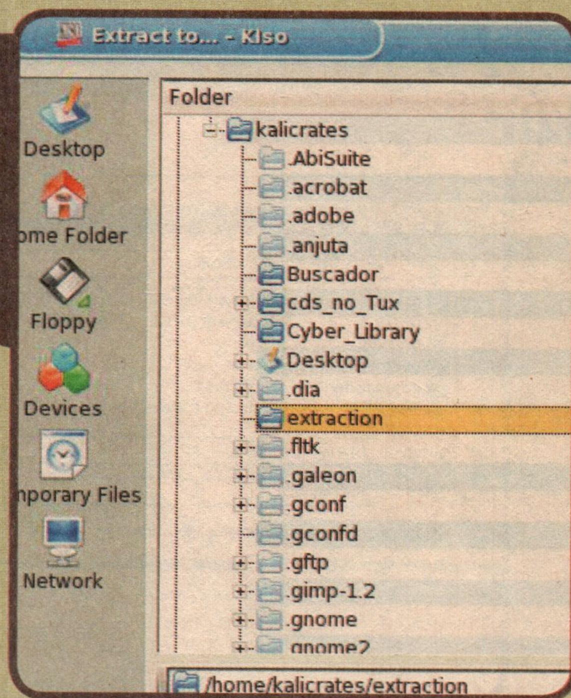


Na janela que será aberta, basta setar um diretório de sua escolha. Nesse caso, selecionamos o diretório "**sos**". Agora, a imagem editada aparece como na tela 17 do nosso passo-a-passo.



Caso você queira salvar a imagem novamente, basta repetir o procedimento já citado. Observe que editamos uma imagem bootável. Como **NÃO** modificamos em nada o diretório que contém a imagem de boot, basta salvar, gravar e bootar com o novo CD para que tudo funcione corretamente.

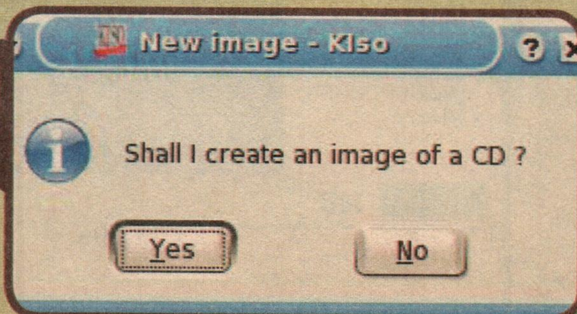
Note que, na imagem, temos 2.86 MB em nosso arquivo ISO editado. Apenas para constar, o diretório sos/ contém uma cópia do arquivo win.iso.



Podemos também extrair uma imagem ISO. Para isso, basta abrir uma imagem e, na seqüência, clicar em **CD Image, Extract image**. O nosso teste está ilustrado na próxima imagem.

É importante lembrar que podemos criar uma imagem de CD diretamente no KISO. Ou seja, utilizando apenas a interface vamos adicionar diretórios ou arquivos, navegar pelos diretórios e também adicionar novos diretórios ou arquivos.

Para isso, clique na seqüência **CD Image, New Image**. Surgirá uma janela como a que se segue:

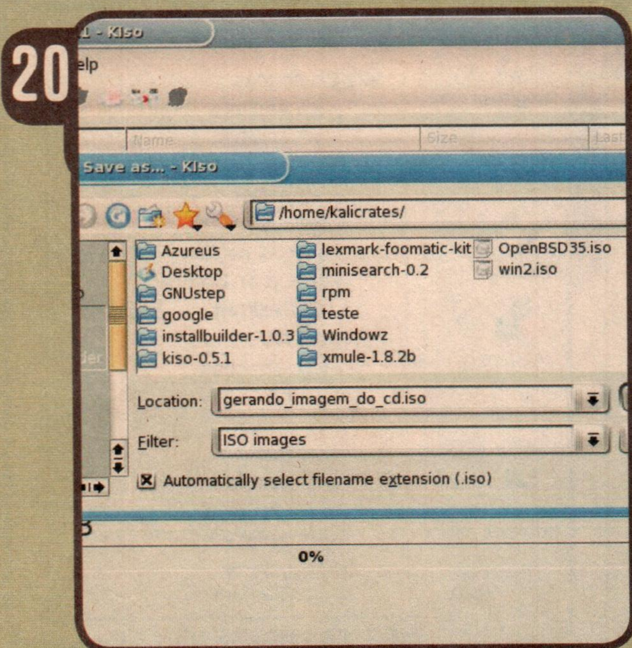


Aqui temos duas opções:

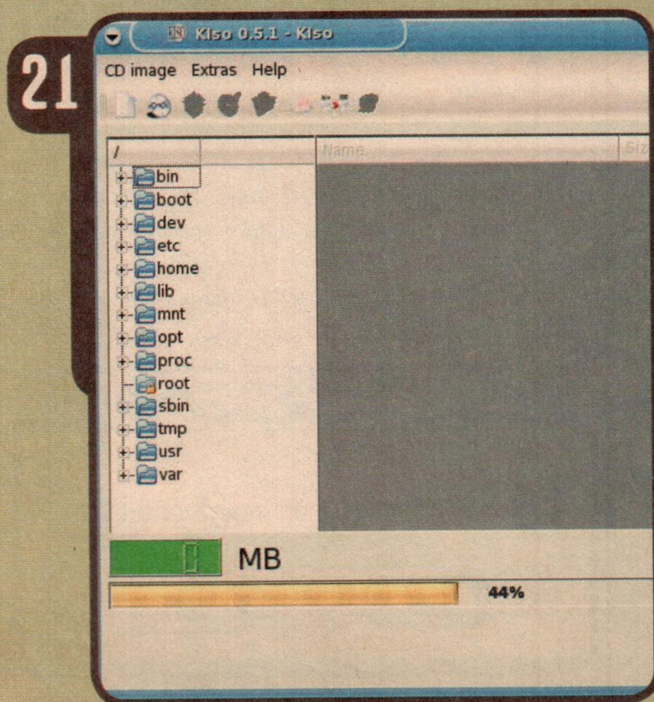
• **Opção No:** nesse caso, criaremos uma imagem ISO a partir de arquivos disponíveis no HD. A área de trabalho do KISO será limpa. Depois, vá adicionando os diretórios ou arquivos e montando a árvore de diretórios que desejar; finalmente, repita os procedimentos para gerar a imagem ISO associada.

• **Opção Yes:** com essa opção, geramos uma imagem ISO a a partir de um CD pré-gravado. É o que vemos nas imagens a seguir:

20



21



GCOMBUST

O Gcombust (**www.abo.fi/~jmunsin/gcombust**) é mais uma GUI para os aplicativos de gravação de CDs em modo texto já discutidos.

Feito em C, o Gcombust utiliza o toolkit GTK+ e, apesar das ressalvas de seu desenvolvedor, Jonas Munsin, de que a versão mais recente é voltada para desenvolvedores, não encontramos grandes problemas nos testes realizados por nossa equipe.

Entre os recursos que o Gcombust nos oferece, temos:

• cópia de CD para CD on the fly, ou seja, sem a geração de uma imagem ISO intermediária no HD
• criação simples de imagens ISO
• suporte a CDs bootáveis

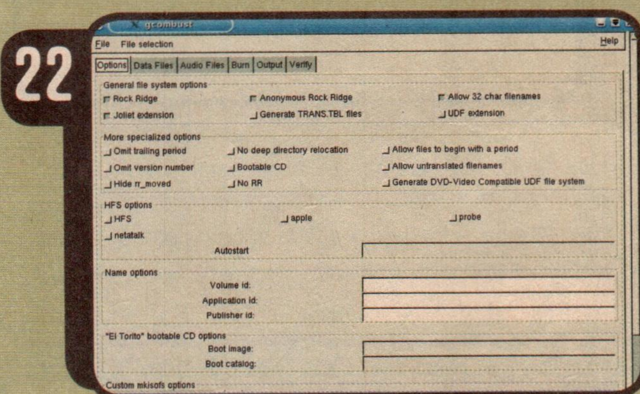
A versão que compilamos é a 0.1.55 (**http://www.abo.fi/~jmunsin/gcombust/gcombust-0.1.55.tar.gz**).

Entre as dependências, temos o **Cdlabelgen**, um aplicativo voltado para a geração de capas de CDs.

As demais são as usuais **Cdrecord**, **Mkisofs**, **Cdparanoia** e **Cdrdao**.

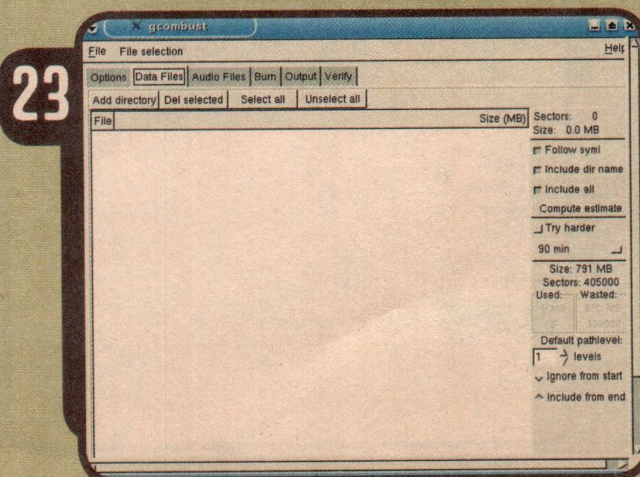
Vejamos algumas imagens capturadas nos testes.

22



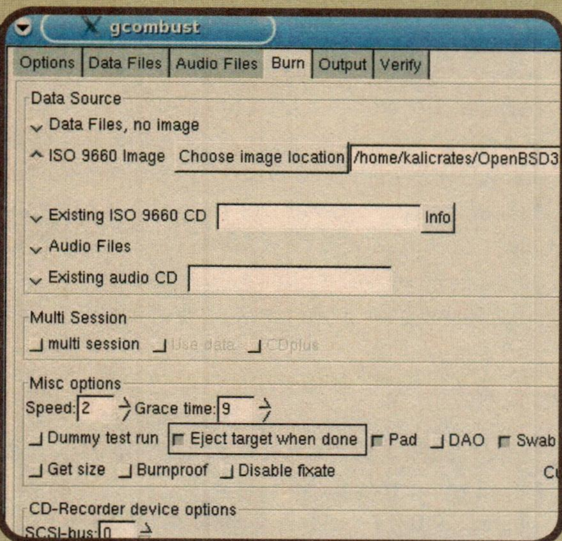
Comentários: essa é a tela principal do Gcombust. Notamos que há muitas opções que podem ser setadas, como extensões **Rock Ridge** e **Joliet**, permissão para nomes de até 32 caracteres, CD bootável etc.

23



Comentários: nesta opção de configuração, temos algo que pode ser muito útil: é possível ajustar a capacidade da mídia de acordo com a nossa preferência (em nosso caso, setamos um CD de 90 minutos com 791 MB).

Se você tiver disponível um gravador de DVDs, basta setar a capacidade adequadàmente.

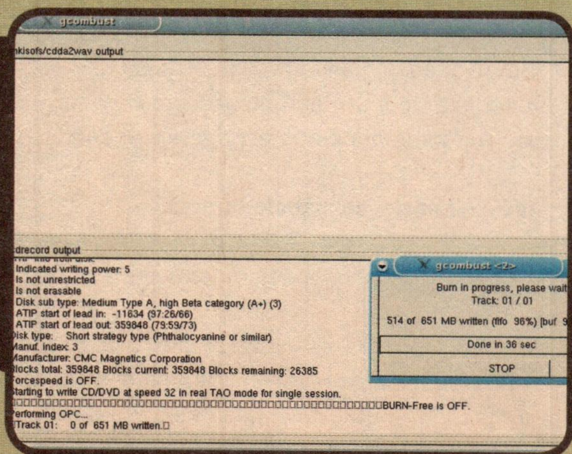


Aqui podemos configurar várias opções para gravação, como **Dummy test** (apenas para simular a gravação), **Eject**, **Overburn** (para ir além dos limites de capacidade padronizados da mídia) etc.

Estas opções refletem os recursos dos utilitários de linha de comando, como já vimos anteriormente em **Data files, No image** e **ISO 9660 image**. Em nosso caso, setamos para que a gravação de uma imagem ISO seja realizada.

Quando tudo estiver configurado, basta clicar em **Combust!** O andamento de uma gravação está ilustrado na imagem exibida na tela 30.

A seguir vemos o Gcombust em ação!



INSTALANDO PACOTES NO LINUX

Você deve ter notado que, em diversas etapas do processo, instalamos aplicativos a partir do código-fonte. Apesar de todo o avanço dos sistemas Unix (o Linux é um exemplo), a grande portabilidade de aplicativos que temos hoje em dia deve-se à própria portabilidade do código-fonte. Portanto, vamos rever essa questão, começando dos primórdios do Unix até o Linux moderno.

NO INÍCIO ERAM OS HACKERS

O Unix é um produto de hackers, mais exatamente da equipe de Dennis Ritchie, que trabalhava nos laboratórios Bell no final da década de 60.

Tendo a C como a linguagem de programação do sistema, os projetistas logo se defrontaram com o problema da compilação de aplicativos para o seu sistema operacional.

No início, o desenvolvedor precisava definir para o ambiente todas as informações, para que a compilação fosse feita corretamente. O motivo para isso é simples. Em geral, um aplicativo útil é um programa complexo, com muitos componentes, que exige um gerenciamento adequado, não só da sua compilação, como da combinação desses itens. Não é difícil perceber que os desenvolvedores Unix suavam a camisa para criar os seus programas.

A saída para esse problema foi a criação de aplicativos que automatizaram o processo de compilação em um ambiente Unix. Esses aplicativos são:

- **Configure:** aplicativo que faz um levantamento do software instalado no seu sistema Unix. Seu objetivo é determinar se todos os recursos necessários (outros aplicativos e bibliotecas de sistema) para a compilação de um pacote estão presentes.

  Ao satisfazer todas as necessidades, o Configure faz a alteração do arquivo **Makefile,** utilizado pelo comando **make**.

- **Make:** do inglês "crie", o aplicativo make toma o script contido no arquivo Makefile e o segue à risca, compilando e linkando tudo que for necessário para que todos os componentes do aplicativo funcionem harmoniosamente.

  Se o comando make chegar ao final sem problemas, falta apenas mais um passo para instalarmos o nosso aplicativo.

- **Make install:** novamente do inglês "crie a instalação", o Make install acessa o arquivo Makefile e

copia para os diretórios corretos todos os executáveis e/ou bibliotecas gerados pelo make, inclusive criando os links simbólicos ("atalhos"), caso sejam realmente necessários.

Mas como os pacotes-fonte são distribuídos atualmente no mundo Unix e Linux?

Há dois formatos principais de distribuição. A seguir, vamos descrevê-los e explicar como devemos proceder para utilizar os comandos de compilação que acabamos de mostrar.

PACOTES .TAR.GZ

Este é um dos tipos de pacote mais comuns. Em geral, trata-se de um diretório, que foi estruturado e compactado. Portanto, antes de trabalhar com o pacote, é preciso descompactá-lo.

Para isso, em primeiro lugar, abra um terminal e digite o seguinte comando:

```
tar -zxvf nome_do_pacote.tar.gz
```

O que significa o código "-zxvf" ?

- **z**: diz ao comando tar para descompactar o pacote; com isso, a extensão .gz (compactada pelo GNU zip) será eliminada;

- **x**: faz com que o comando tar extraia os arquivos do pacote para o diretório que será criado;

- **v**: modo verbose que é uma opção comum nos utilitários Unix. Diz ao comando tar para descrever a evolução da descompactação à medida que este processo for realizado;

- **f**: descompacta o pacote na saída usual.

Voltando ao nosso problema da compilação de um pacote, o comando tar descrito irá gerar um arquivo com o nome **nome_do_pacote**, preservando toda a sua árvore de diretórios (as suas pastas).

Agora, basta acessar o pacote e utilizar os comandos de compilação, na seguinte seqüência:

```
cd nome_do_pacote /* acessa o diretório gerado
*/
./configure /* faz o levantamento de dependências
*/
make /* compila os aplicativos e bibliotecas ne-
cessários */
```

Depois desse último passo, resta apenas instalar o aplicativo. Nesse ponto, é preciso ressaltar alguns detalhes: o Unix é um sistema que possui um usuário singular. O root (ou superusário ou administrador) é um usuário que detém poder total no sistema, com o direito de incluir e/ou remover usuários, apagar qualquer diretório, ou até mesmo destruir o sistema. Devemos logar como usuário root somente quando for estritamente necessário. Para esse usuário privilegiado, é reservado o direito de instalar aplicativos que possam ser usados por qualquer usuário. Portanto, o passo final para instalarmos nosso aplicativo é mudar o nosso status de usuário para root e usar o comando make install.

```
su /* muda o nosso status de usuário para root
– será requisitada uma senha */
make install /* instala o aplicativo! */
```

Recapitulando, a seqüência de compilação/instalação de um pacote-fonte é:

```
tar -zxvf nome_do_pacote.tar.gz
cd nome_do_pacote
./configure
makefile
su
make install
```

PACOTES .TAR.BZ2

Em relação ao caso anterior, a única diferença no procedimento é um parâmetro para descompactar o arquivo, como podemos ver neste comando:

```
tar -jxvf nome_do_pacote.tar.bz2
```

Depois, a seqüência de comandos é a mesma.

Mas nesse processo todo ainda há um problema muito chato de se resolver. Depois de instalarmos o aplicativo, o que devemos fazer quando for pre-

ciso desinstalá-lo?

Às vezes, os desenvolvedores incluem um arquivo de desinstalação, que pode ser invocado pelo seguinte comando (disparado a partir do diretório gerado):

```
make uninstall
```

E se apagamos o diretório no qual realizamos a compilação?

No caso do Linux, já existe uma alternativa muito interessante: trata-se do aplicativo **Checkinstall** do coder Felipe Eduardo Sánchez Díaz Durán (**http://checkinstall.izto.org**).

Quando invocado no lugar do make install, ele gera um pacote nativo de uma das três principais distribuições do mundo Linux (explicaremos o que são estes pacotes mais adiante):

.rpm (Red Hat), .deb (Debian) ou .tgz (Slackware)

Depois de instalarmos o Checkinstall, a seqüência de compilação e instalação toma a seguinte forma:

```
tar -zxvf nome_do_pacote.tar.gz
cd nome_do_pacote
./configure
makefile
su
checkinstall
```

Após essa etapa, o Checkinstall fará algumas perguntas muito simples, por exemplo, "qual sistema de pacotes você deseja", "um breve comentário sobre o pacote a ser criado" etc.

Mas, afinal de contas, qual é a vantagem de se usar o Checkinstall?

A resposta é simples. Teremos, ao final de todo o processo, um pacote ajustado à nossa distro, que pode ser manipulado facilmente pelos gerenciadores de pacotes, cujo conceito veremos logo a seguir.

GERENCIADORES DE PACOTES

O que é um pacote binário no Linux? Os desenvolvedores Linux pensaram em compilar um aplicativo e gerar um arquivo contendo tudo o que fosse necessário para que o usuário pudesse instalar o que desejava. Com um pacote binário já pronto, usamos apenas algumas regras para instalá-lo (sempre como um usuário root).

Os principais formatos de pacotes e os comandos de controle básicos são os seguintes:

Pacotes .rpm

Formato nativo da distribuição Red Hat, atualmente também usado pelas distros SuSE e Conectiva.

· Instala opção de pacotes

rpm -ihv nome_do_pacote.rpm

Em que:

i: diz ao gerenciador rpm para instalar o pacote

h: mostra uma barra de evolução do processo de instalação no terminal

v: modo verbose, no qual o processo de instalação é comentado durante a sua execução

· Remoção de pacotes

rpm -e nome_do_pacote

Em que:

e: informa ao gerenciador rpm a necessidade de desinstalar o pacote

· Atualização de pacotes

rpm -Uhv nome_do_pacote.rpm

Em que:

U: informa ao gerenciador rpm que se trata de uma atualização

Pacotes .deb

Sistema de pacotes criado pelos desenvolvedores da distro Debian

· Instalação de pacotes

dpkg -i nome_do_pacote.deb

Em que:

i: informa ao gerenciador de pacotes que se trata de uma instalação

· Remoção de pacotes

dpkg --purge nome_do_pacote

Em que:

purge: indica ao gerenciador que, tanto o aplicativo quanto todos os diretórios e arquivos de configuração gerados devem ser removidos.

Pacotes .tgz

Esse é o último sistema de pacotes que vamos apresentar. O .tgz é o formato dos pacotes compilados da distro Slackware. Seus comandos de controle são os seguintes:

· Instalação de pacotes

installpkg nome_do_pacote.tgz

· Remoção de pacotes

removepkg nome_do_pacote.tgz

· Atualização de pacotes

upgradepkg nome_do_pacote.tgz

Mas tudo são flores? Infelizmente, não. O processo de instalação descrito não é completamente automático porque, ao fornecer um aplicativo pronto (o pacote!), pode acontecer de o usuário não ter todos os componentes (outros aplicativos e bibliotecas) necessários para que o programa funcione. Isso pode ser evitado, na maioria dos casos, quando realizamos uma compilação nativa.

Para controlar o processo, foram criados gerenciadores de pacotes mais poderosos, que se utilizam de uma base de dados e de pacotes locais ou remotas. Eles controlam as dependências, de forma que, no final do processo, temos realmente um processo automático para instalação de pacotes.

Os principais gerenciadores são os seguintes:

Apt-get

O primeiro e, provavelmente, o mais poderoso dos gerenciadores de pacotes do mundo Linux. Criado pelos desenvolvedores da distro Debian, pode ser utilizado facilmente, como vemos a seguir:

apt-get update /* atualiza a lista de pacotes */
apt-get install nome_do_pacote /* baixa e instala o pacote em questão, resolvendo todas as dependências */

Swaret

Este é um projeto mais recente (**www.swaret.org**), cujo intuito é resolver um problema antigo da distribuição Slackware: o gerenciador básico do Slack simplesmente não controla dependências. Depois de instalado e configurado, bastam poucas linhas de comando para resolver esse problema:

swaret --update /* atualiza a lista de pacotes */

Para instalar um aplicativo:

swaret --install nome_do_pacote

Para atualizar um aplicativo:

swaret --upgrade nome_do_pacote

Com isso, terminamos nosso breve apêndice sobre instalação de programas no Unix/Linux, mas devemos frisar que há muito mais para aprender. Indicamos a leitura da documentação da versão do Unix/Linux que você utiliza para mais informações.*Gk*

REFERÊNCIAS BIBLIOGRÁFICAS:



1) MORIMOTO, Carlos Eduardo. *Entendendo e Dominando o Linux.* São Paulo: Digerati Books, 2004. p. 59 a 61.

2) Páginas de Manuais do Cdrecord, Cdrdao, Mkisofs.